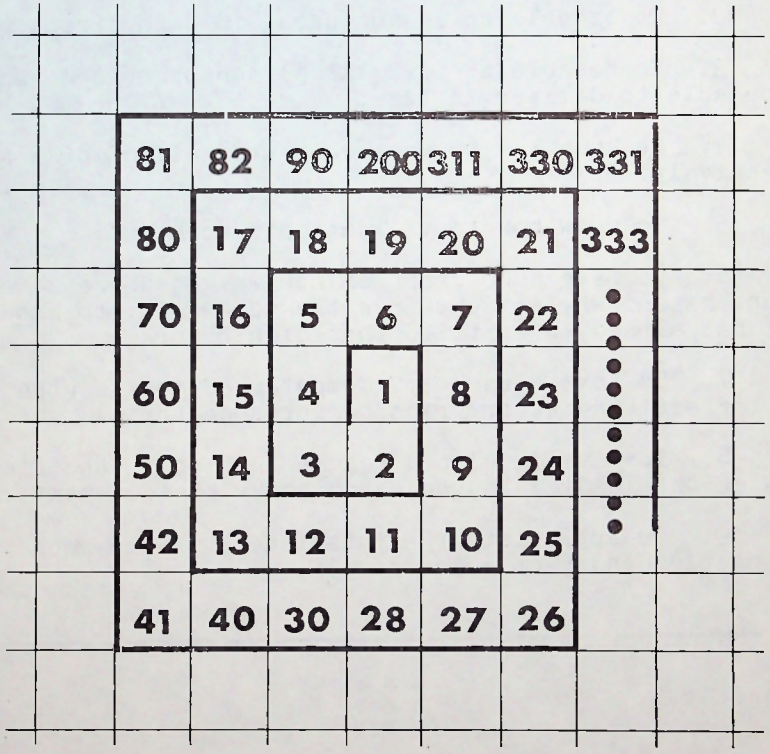
$2.50

66

September 1978 Volume 6 Number 9

Popular Computing

The only magazine for people who enjoy computing.



See Problem 240 page 5

The Chicago Loop Trip

What Is "Computer Science"?

The computing student should know what it is that industry expects of him as a result of his academic training. It boils down to this:

1. To be able to code, of course, in some high level language, plus some familiarity with machine and assembly language.

2. To be able to communicate, in both directions.

3. To be able to work largely unsupervised, and to be able to deliver on time.

4. To know how to use a computer efficiently and effectively.

5. To know how to validate his work.

6. To have some sense of how we got where we are; to understand why we do things the way we do; to know what has already been tried that didn't work.

7. To have a sense of computer economics (that is, cost/effectiveness, or price/performance ratio).

8. To have developed reading habits in the field, both of the leading journals and books besides texts.

9. To appreciate the interfaces between machines and people, and people and people.

Publisher: Audrey Gruenberger
Editor: Fred Gruenberger
Associate Editors: David Babcock
Irwin Greenwald
Contributing Editors: Richard Andree
William C. McGee
Thomas R. Parkin
Advertising Manager: Ken W. Sim
Art Director: John G. Scott
Business Manager: Ben Moore

POPULAR COMPUTING is published monthly at Box 272, Calabasas, California 91302. Subscription rate in the United States is $20.50 per year, or $17.50 if remittance accompanies the order. For Canada and Mexico, add $1.50 per year. For all other countries, add $3.50 per year. Back issues $2.50 each.

Question A

What constitutes a problem suitable for computer solution? This is Question A.

Let us keep in mind that the use of computers to solve problems dates from around 1950, and at that time the world had already produced such things as atomic bombs, jet engines, electronic television, and marvelous and extensive mathematical tables. Clearly, a broad spectrum of complex problems can be solved with analytical methods, graphical methods, mechanical (unsequenced) calculators, punched card equipment, and paper-and-pencil.

If we view the computer simply as a fast, large, and reliable sequenced calculator, then the answer to Question A is this: any problem that is too large or too involved to be carried out, in limited time, by human effort. In theory, any processing of information (within certain loose constraints) can be done by manual or mechanical methods, given enough time and determination. Baron von Vega's 7-place logarithm table was calculated over a hundred years ago.

But looking at it another way, there must clearly be a limit to what can be handled without a computer (loaded, of course, with the proper and correct program). At some point a difference in degree becomes a difference in kind.

Let's expand on that theme for a bit. We learned in algebra that

$$(a + b)^2 = a^2 + 2ab + b^2$$

so that we had a model we could paraphrase to square any binomial. Thus:

$$(3x + 7y)^2 = 3^2x^2 + 2\cdot3\cdot7\cdot x\cdot y + 7^2y^2$$

$$(3.123x + .00765y)^2 = 3.123^2x^2 + 2\cdot3.123\cdot.00765xy + .00765^2y^2.$$

If we had a good algebra course, this handy plan was extended in two directions: to higher powers, and to polynomials of more than two terms. Thus we could have paradigms for:

$$(a + b)^3 = a^3 + 3a^2b + 3ab^2 + b^3.$$

$$(a + b)^4 = a^4 + 4a^3b + 6a^2b^2 + 4ab^3 + b^4.$$

$$(a + b + c)^2 = a^2 + b^2 + c^2 + 2ab + 2ac + 2bc.$$

$$(a + b + c)^3 = a^3 + b^3 + c^3 + 3(a^2b + a^2c + b^2a + b^2c + c^2a + c^2b) + 6abc.$$

And this collection of models could be carried to any desired limit. Thus, in theory, there is no real difference between these expansions:

$$(3.123x + .00765y)^2$$

$$(3.123x + .00765y - 543.21z)^{75}$$

except that the second one involves a lot of paper, work, and time, and has a high probability of being calculated wrong. Going just a little bit further, when we need the quotient:

(B)
$$\frac{(3.123x + .00765y - 543.21z - 73\,w)^{181}}{(5.8x - 12.71y + 888.8z + (5/13)w)^{97}}$$

the chances of achieving the correct quotient and remainder by hand methods become rather slim. Moreover, the prospect of doing such a tedious and boring task becomes too grim to bear. Have we, then, arrived at an example answer to Question A? The manipulations for fraction (B) simply require patience, dogwork, and storage--all attributes that the computer has in abundance.

But surely the computer offers us more than the ability to be a tireless slave. "The computer lets us play out the consequences of our decisions." Among other things, this means that we can at least begin a problem solution with an inefficient (if not downright stupid) algorithm in order to acquire some results; frequently we can see a pattern in a few results and be led to a better algorithm. Or, we are privileged to try several attacks on one problem and enjoy the luxury--seldom possible in pre-computer days--of wild guessing, deliberate blind alleys, and unrestricted imagination in our approach to a new and baffling situation.

Over the years, we at POPULAR COMPUTING have frequently experienced the same phenomenon; namely, what we thought was a beautiful computing problem that is completely demolished by clever analysis or even by manual methods. Perhaps the best (or worst) example of this occurred with our Contest 15 (see issue No. 57, December 1977), for which we devised a problem that could only be solved by manipulating huge masses of data--only to find that the contestants chose to exercise great analytic skill and carefully avoided all such manipulation. However, in every case the clever analytic solution required the use of a computer to carry out the details (which involved very large numbers), thus furnishing a splendid example of synergy.

It is my contention that the computer is a new tool in the service of man; not a larger, speeded-up, and transistorized old tool, but something really new. If that contention is true, then it seems apropos to look for the dividing line between those things that could have been done (perhaps with difficulty) prior to the computer age, and those things that uniquely belong to the computer age. To that end, it does not seem to me to be necessary to justify every problem. If we are interested in the art of computing, then any computing problem is a fair challenge, and the good ones are simply the ones that interest you. Some 240 new problems have been presented in our pages, with sufficient variety that each person can find some that he enjoys working on. The point is, it is difficult to establish, a priori, that a given problem <u>is</u> a computing problem.

Consider, for example, the problem illustrated in the Figure (C) on the cover of this issue. We start at the center of a spiral of squares, and insert consecutive integers in each cell, subject to one simple restriction: adjacent cells (horizontally or vertically, but not diagonally, and with reference only to squares on an inner ring, not to squares just prior to the one being entered) should not contain repetitions of the same decimal digit. Thus, the numbering is consecutive up to 28; a 29 in the next cell would conflict with the 12 in the cell just above it (i.e., duplicate 2's), so the next possible integer is 30, as shown. Similarly, for the next cell, any number in the 30's is inadmissable because of the 3 in 13. Thus the sequence shown on the cover develops; the Northeast diagonal elements are 1, 7, 21, 331,...

Now, suppose we wish to have the next ten elements along that Northeast diagonal--is that a computer problem, or could it be done without a computer? The solution requires filling 660 more cells, and it would be difficult even to guess the order of magnitude of the numbers at the end. (I think that the next number in the sequence along the diagonal is 2201, but I haven't verified that calculation with a computer program.)

PROBLEM 240

Consider another problem situation. Take a 10 x 10 array of cells, whose contents are initially 2-digit numbers from 00 to 99 as shown by the bold figures in Figure D. The units digit of these initial numbers is thus the column number from zero to 9 and the tens digit is the row number from zero to 9.

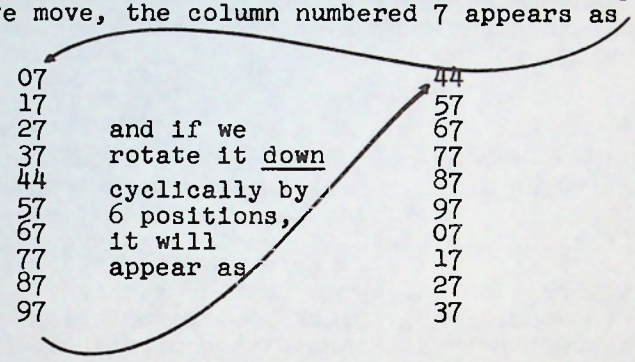
We arrange to be able to shift any row or column cyclically by any amount from 1 to 9 positions. For example, row number 4 initially has the contents:

40 41 42 43 44 45 46 47 48 49

and if we rotate it cyclically to the right 3 positions, it will then contain the numbers:

47 48 49 40 41 42 43 44 45 46.

Similarly, we can select any column and rotate it cyclically down by any amount from one to 9 positions. For example, after the above move, the column numbered 7 appears as



Let us denote by the 3-digit number xyz the following:

x = 0: horizontal shift
1: vertical shift

y = row or column number (0 to 9)

z = amount of shift, to the right or down.

(The two shifts previously described could thus be dictated by the numbers 043 and 176.)

Then if we wish to reverse the northwest-southeast diagonal of the array, the following sequence of moves (among others) will do it:

009, 017, 025, 033, 041, 059, 067, 075, 083, 091,
101, 113, 125, 137, 149, 151, 163, 175, 187, 199.

This brings us to the point: Could the above operations be carried out by hand correctly, and could we predict what will happen to all the other cells of the array? All of this is shown in Figure D, but the operations were carried out by a computer program.

00	01	02	03	04	05	06	07	08	09
99	76	53	30	17	94	71	58	35	12
10	11	12	13	14	15	16	17	18	19
01	88	65	42	29	06	83	60	47	24
20	21	22	23	24	25	26	27	28	29
13	90	77	54	31	18	95	72	59	36
30	31	32	33	34	35	36	37	38	39
25	02	89	66	43	20	07	84	61	48
40	41	42	43	44	45	46	47	48	49
37	14	91	78	55	32	19	96	73	50
50	51	52	53	54	55	56	57	58	59
49	26	03	80	67	44	21	08	85	62
60	61	62	63	64	65	66	67	68	69
51	38	15	92	79	56	33	10	97	74
70	71	72	73	74	75	76	77	78	79
63	40	27	04	81	68	45	22	09	86
80	81	82	83	84	85	86	87	88	89
75	52	39	16	93	70	57	34	11	98
90	91	92	93	94	95	96	97	98	99
87	64	41	28	05	82	69	46	23	00

D

A program for the PeCos One personal computer was written by Associate Editor Irwin Greenwald to perform such manipulations on the 10 x 10 array. With Greenwald's program available, it is feasible and relatively easy to explore the solution to a wide range of problems, such as those listed on the next page.

The solution to these problems is not important at the moment (although they can be interesting by themselves)--we are back to Question A:

How many such problems could be solved without a computer?

1. Could the NW-SE diagonal be reversed in fewer than 20 moves?

2. What set of moves will reverse the SW-NE diagonal (the one whose initial contents are 90, 81, 72, 63, 54, 45, 36, 27, 18, and 09)?

3. Is there a set of moves that will reverse both diagonals at once?

4. If the NW-SE diagonal elements are reversed, and then reversed again with the same sequence of moves, and so on, eventually the original ordering of the entire array will be restored. How many reversals will it take?

5. What is the minimum set of moves to interchange the contents of two cells?

6. What set of moves will interchange a row with a column? For example, what set of moves will turn row 5 into column 8 and column 8 into row 5 at the same time?

7. Can <u>any</u> arbitrary arrangement of the numbers be achieved, or is there some set of arrangements that are impossible to arrive at from the initial ordering?

Here's another problem, this one from the <u>Journal of Recreational Mathematics</u>. Let's see if it qualifies as a computer problem:

$$a_0 = 0$$

$$a_1 = 1$$

$$a_{n+1} = a_n - \left[\tfrac{1}{2}(a_n + 1)\right]$$

unless that number has occurred earlier, in which case the sign changes from - to +.

The square brackets denote "greatest integer in," so the function is, roughly, one-half the previous term unless that one has appeared already, in which case it's one-and-one-half the previous term. It is certainly an easy function to calculate; Figure E is a flowchart of a possible scheme. The first few terms are:

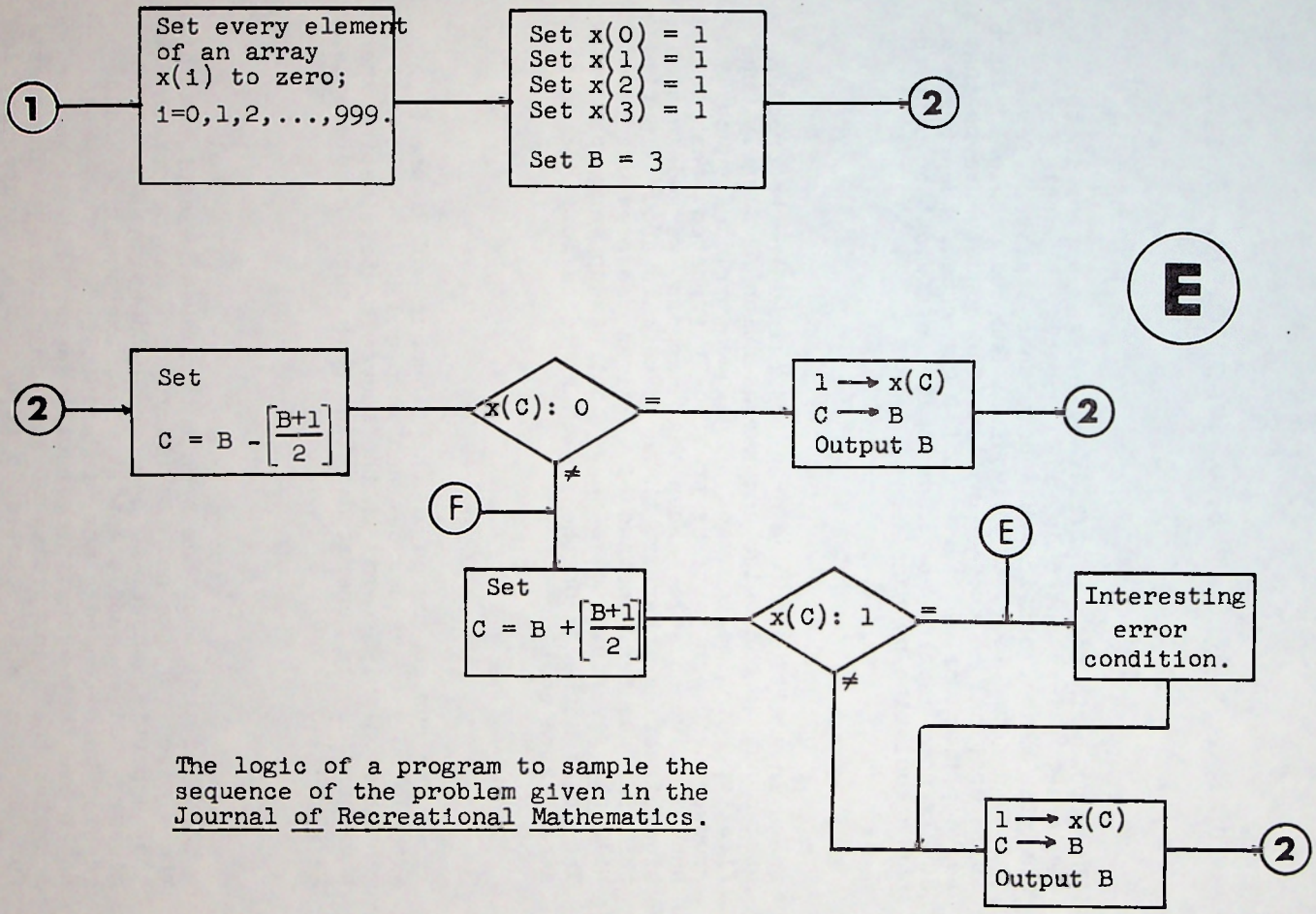
0, 1, 2, 3, 5, 8, 4, 6, 9, 14, 7, 11, 17, 26, 13,...

The problem is this: will every integer appear in this sequence? That question will probably be settled by analytic means, but it would help to calculate a few thousand terms, say, to get a feel for the problem. Since it is critical to the generation of successive terms that we keep track of all terms so far generated, then there will surely come a point at which it is inefficient to continue the calculation by hand.

In addition to the question raised in the problem statement, we can raise another question. Clearly, the first try at a new term of the sequence will frequently produce a number that has previously appeared (that is the nub of the problem). When that happens, we add instead of subtracting; that is, the next term is about 1.5 times the previous term instead of .5 times the previous term. Will <u>that</u> ever produce a value that has appeared before? The point marked E on the flowchart tests for that case at no extra cost.

The run suggested in the flowchart of Figure E extends only to values between zero and 999. It will generate only 190 terms in the sequence, terminating when the term 1214 is generated. The path through Reference F is taken 123 times.

Thus, not much has been established, except that in the limited run the path to Reference E was not taken, so that a partial answer to one of our questions has been given. Given sufficient storage, the sequence could readily be extended almost indefinitely. This would not prove that every integer will appear, but it would increase the plausibility of the conjecture. The question for us is: at what point does it become a computer problem?



The logic of a program to sample the sequence of the problem given in the Journal of Recreational Mathematics.

Consider now a 5th problem that might help us to decide where the dividing line is between computer problems and non-computer problems. See Figure F.

A grid of squares, 12 x 12, is filled with 3-digit random numbers by the following procedure. We use Dr. Schwartz's random number generator:

$$\text{new } x = \text{fractional part of } (\text{old } x + \text{pi})^5$$

with the initial value of old x taken as zero. Thus, the generator will furnish random numbers between zero and one, and we will take the greatest integer in

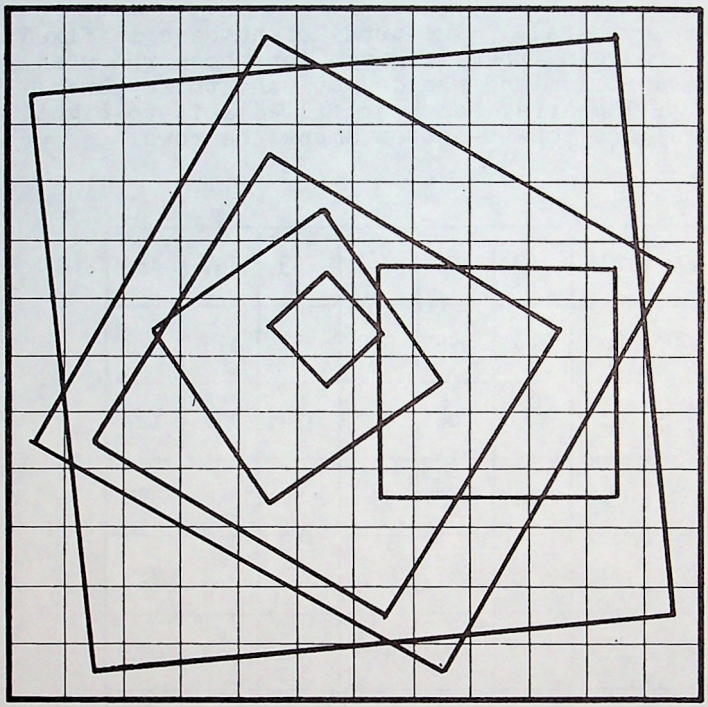
1000 times x

for our 3-digit numbers. The array of numbers is fixed; the scheme of producing them is given for those who wish to work on this problem and who do not want to keypunch 1000 numbers. The array has been filled left-to-right across the rows, and top-to-bottom among the rows.

019	727	581	133	897	005	675	496	763	193	797	438
349	669	795	256	300	280	381	049	030	257	579	643
009	721	201	835	250	417	713	806	857	724	151	127
326	623	693	149	321	051	121	889	991	482	346	587
067	486	695	688	755	362	453	257	963	573	412	719
350	346	437	279	692	999	068	989	496	281	652	092
657	135	152	695	292	884	764	641	977	628	309	689
783	512	472	987	306	635	294	073	436	630	198	942
847	477	182	095	358	572	221	947	202	165	607	674
743	586	428	336	140	127	582	127	259	132	241	124
790	865	546	668	325	026	917	796	432	406	452	790
616	033	789	987	551	400	253	322	921	851	935	830

On this grid, there are combinations of four squares whose connected centers themselves form squares, as shown by some examples in Figure G. There are 1716 such combinations. For each combination, the four squares that define the vertices determine four numbers, and these four numbers have a sum, S, and a product, P. For two distinct combinations of four squares, A and B, there is a ratio:

$$\frac{S_A \cdot P_B}{S_B \cdot P_A}$$



G

and for some two combinations that ratio is

a) smaller
b) larger

than for any other combinations on the grid. Find the A and B combinations for the two criteria. Table H shows some sample calculations on this problem. Groups 7 and 9 in the sample will produce ratios of 791.99 and .0012626, which are the largest and smallest that can be achieved with this tiny sample of the combinations that could be produced from the 1716 squares on the 12 x 12 grid.

					Sum	Product ($\times 10^9$)
1	884	068	294	641	1887	11.328347
2	095	358	140	336	929	1.599830
3	797	438	643	579	2457	129.963671
4	019	438	830	616	1903	4.254872
5	727	643	935	790	3095	345.290068
6	133	587	921	847	2488	60.902172
7	009	789	124	193	1115	.169941
8	067	695	152	657	1571	4.650157
9	991	587	977	689	3244	391.584544
10	999	989	073	635	2696	45.799250
11	999	989	889	051	2928	44.795431
12	067	051	221	586	925	.442522

Even that minor result required extensive calculation, although not the use of a computer. Suppose that the given problem could be exhausted by diligent work by hand, or with the use of a programmed calculator. The problem could then be stepped up to a 13 x 13 grid or higher, to increase the complexity enormously. There must be some point at which methods short of computer use are doomed to failure--and we are back to Question A.

Can we reach any conclusion from all this? The five illustrative problems all suggest that we consider the matter of worthwhileness. Each of the problems so far presented is intrinsically useless, and hence could be regarded as not worth considering, which means not worth computing. If you follow that line of reasoning to its logical conclusion, then only useful and worthwhile problems should ever be submitted to a computer. But useful to whom?; worthwhile for whom? Recall that binary arithmetic was labelled as novel but useless as recently as 1940; that up to 1960 our industry was still trying to make computers that operated in decimal; that four times in our short history there have been attempts to make machines that would operate directly on fractions.

It would seem wise to use great care and caution before relegating any given problem to the "worthless" pile. At a bare minimum, almost any problem can be considered useful and worthwhile as a device to improve our problem solving capability.

In any event, let's play the game. We have before us some problems to be solved. Let's not quibble about their effect on the world; let's just solve them. At what point, then, does the computer change from a handy device to an essential tool? We might try to list some clear-cut cases:

1. Any problem solution that calls for very high precision. This situation was explored at length in the essay "How High the Precision?" in our issue No. 52. The computer is the only device that permits us to function with any arbitrary precision level. For example, if it is required to find all the digits before the decimal point in

$$\frac{{}_{100}C_{80} + {}_{90}C_{10}}{{}_{60}C_{30} - {}_{35}C_{15}}$$

(the notation refers to the number of combinations of N things taken R at a time), then the computer is very likely the only possible tool.

2. Any information processing situation in which the sheer bulk, or severe time constraints, or both, eliminate any possibility of hand or punched card, or calculator methods. Examples: updating the Social Security file each day; accounting for every airplane seat for a large airline for every flight for the next year; cross indexing all the telephones in Chicago, with the cross index up to date each day.

3. Most Monte Carlo problems, in which a problem situation is essentially turned into a game, and the results of playing the game thousands of times gives an approximation to the true solution to the problem.

4. Combinatorial problems. Examples: the Checkerboard problem (in our issue No. 15)--in how many ways can a 14 x 14 checkerboard be cut into four congruent pieces, following the lines of the board? Polyominoes (in our issue No. 22)--how many polyominoes of order 14 are there?

5. The Reordering problems (which started with Problem 193 in our issue No. 55). Rather than review those problems, consider a new one:

Start with all the positive integers beginning with 2. Whatever number is at the head of the stream, reverse that many sets of that many numbers. For example, if the number at the head of the stream is 13, then the first 13 sets of 13 numbers are to be reversed. We seek a list of the numbers that will appear at the head of the stream.

The first few stages of this Reversal problem are shown in Figure J at the left, and indicate that the sequence being sought begins with

2, 3, 5, 6, 11, 9, 2, 3, 6, 7, 14,...

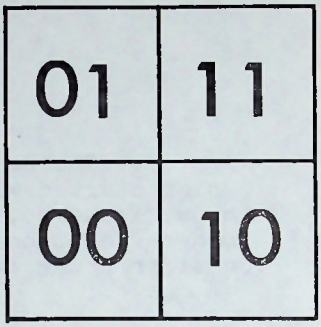
With extraordinary patience, it might be possible to carry this process through 30 stages or so by hand. On the other hand, it is quite likely that someone could find an analytic scheme to extend the list indefinitely.

But if the list could only be extended by actually manipulating the numbers, then to go beyond some point (say, 30 terms) would definitely require a computer, would it not?

Further discussion of Question A will be welcomed.

Fred Gruenberger

Two bits can be used to identify the four quarters of a square as follows:



Suppose we take a proper fraction and express it in binary:

1/3 = . 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 ...

and use each successive pair of bits to subdivide a unit square. The repetitive "01" pattern calls for moving to the northwest quarter, and then to the northwest quarter of that quarter, and so on. The end result converges to the upper left corner of the unit square, and is shown as "3" in the Figure (A). In similar fashion, the "8" in the Figure denotes the result of taking

1/8 = . 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 ...

and following the effect of successive quartering. The lower left corner represents the converging point of the number zero, and the upper right corner represents the number one. All the other numbers in Figure A represent denominators of unit fractions.

The Figure shows that this procedure is clearly not a continuous process. From the given eleven points, there is little clue as to where 1/12 would land. Worse, the values between 9 and 10 (for example) do not give rise to a smooth set of points between the points shown for 9 and 10. In fact, we can calculate the following:

N	X	Y
9.0	.1429	.4286
9.1	.1429	.3810
9.2	.2245	.3664
9.3	.2258	.3226
9.4	.2186	.3341
9.5	.2407	.2975
9.6	.2500	.2500
9.7	.2038	.2914
9.8	.2000	.2804
9.9	.1705	.3659
10.0	.1667	.3333

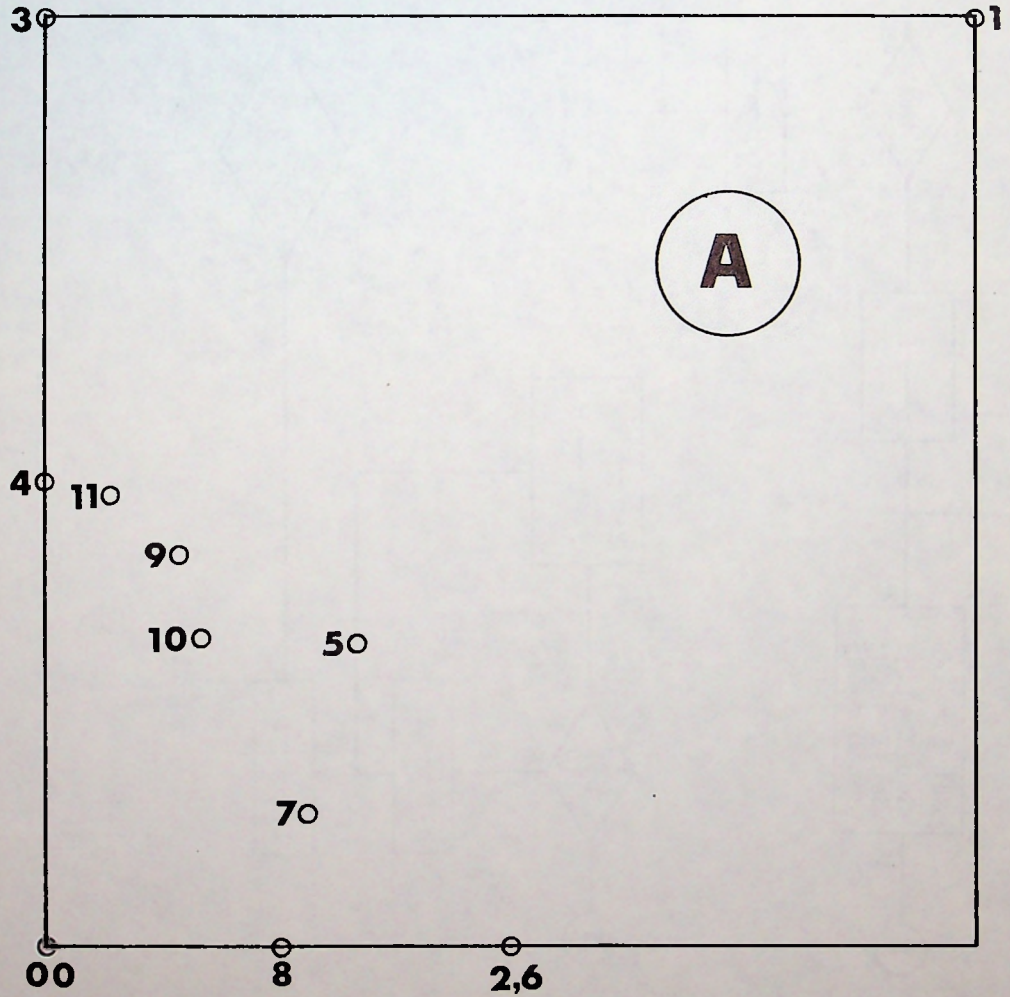
indicating a path that jumps around wildly. Figure B shows the gyrations involved in going from N = 1.480 to N = 1.496. The binary values for 1/N thus go from

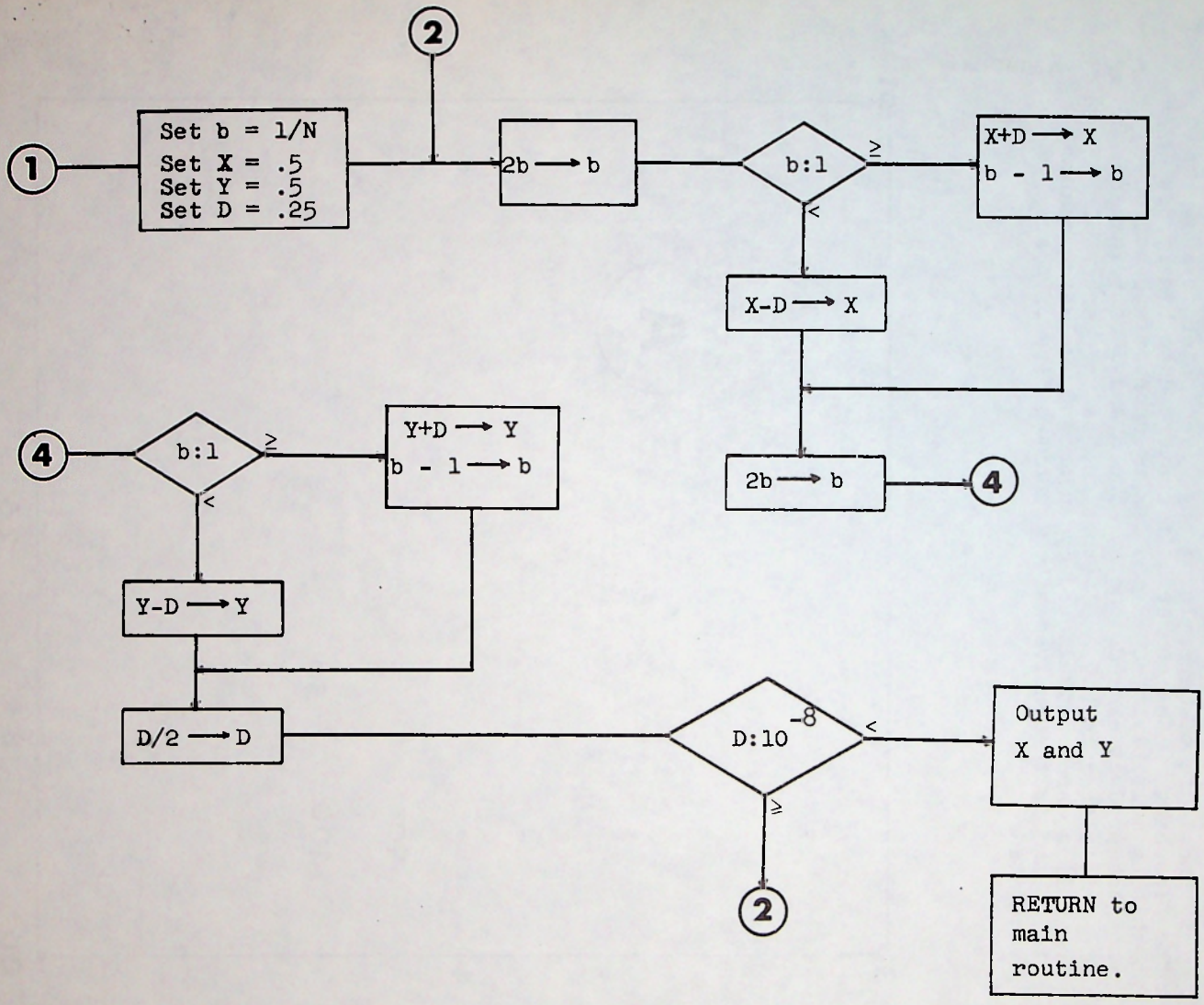
.10101100111110010001010O1...
to .10101011010110100010000011...

and, in theory, should plot as a smooth continuous curve.

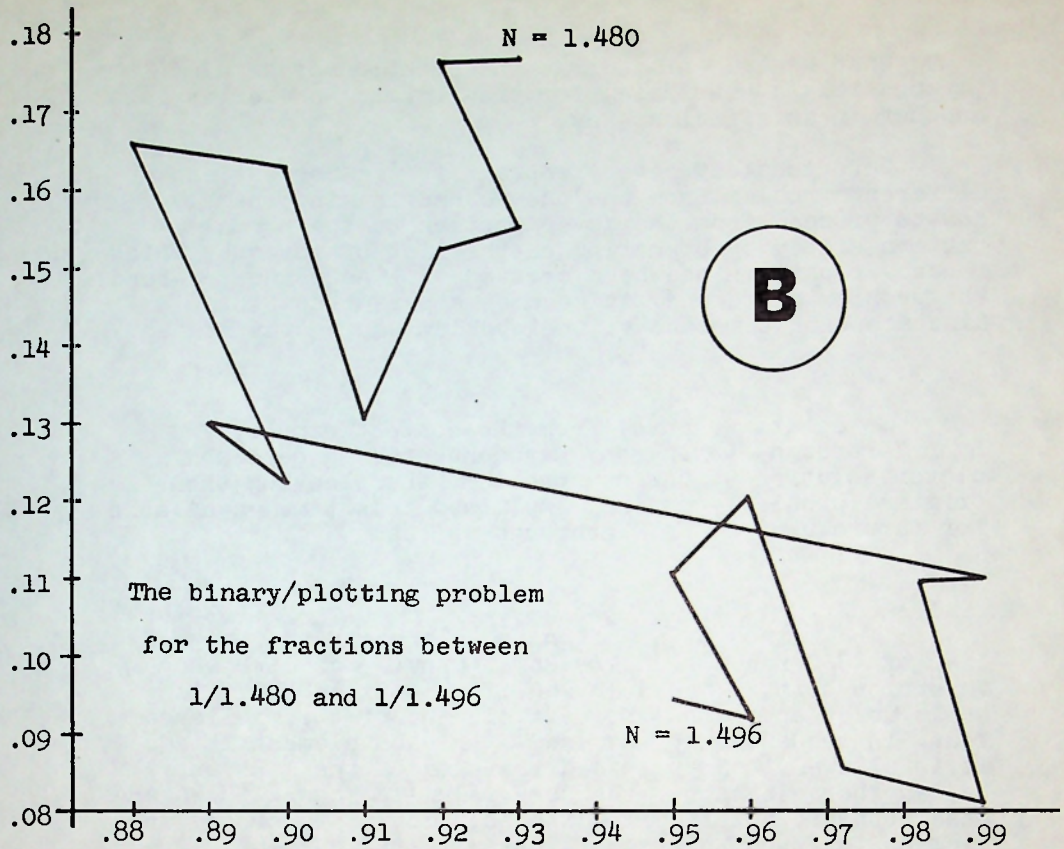
The curious behavior of this function calls for more exploration. It would seem to be ideal for the use of a plotter.

The logic of determining any given point is shown in the flowchart. The number b is converted to binary bit by bit by multiplying by 2 and taking appropriate action when the product exceeds one. The X and Y coordinates are initialized to (.5, .5) (that is, to the center of the unit square) and altered by an amount, D, which starts at .25 and is then halved after each two bits of b are calculated.





A possible scheme for a subroutine for our binary/plotting problem. The value of N is set in the main routine.



Contest Result

Contest 16 (in issue No. 60, March) consisted of the following problem. A 12 x 12 array was filled with the numbers from 1 to 144 in normal left-to-right, top-to-bottom order. Then sets of four adjacent numbers in the array, both horizontally and vertically, were selected at random and their order reversed. This was done for 1000 sets, and the resulting array was presented (on the cover) as the starting array for the contest. The list of the 1000 sets-of-four was given, to show exactly how the cover pattern was created and to prove that the proposed problem was possible. The problem was to go from the given array back to the normal ordering by a sequence of reversals-by-four.

What we had in mind was the creation of an algorithm (or possibly a heuristic) for transforming one array into another in an efficient way.

Unfortunately, every entrant to the contest saw a different problem from the one we had in mind; namely, how to proceed from the cover pattern to the normal pattern simply by operating on the list of moves by which the cover pattern had been created. (As we now re-read the problem statement, it becomes apparent how this misconception came about; the problem was poorly stated.)

So be it; we have, from the contest winner, Mr. Lynn Yarbrough, Lexington, Massachusetts, a neat and clever solution to the new problem (thus leaving the original problem--properly explained this time--available for future use). Mr. Yarbrough writes:

"The method of solution is to process the original list of 1000 random numbers in a manner much like an insertion sort. The solution list, called SHORT, is built up by appending the next element of the published list, in sequence, at the end. As each element is added, it is allowed to move toward the head of the list until (a) another element of the same value is found, in which case both are deleted from the solution list, or (b) an element is found which conflicts with the current element, in that their order cannot be reversed without affecting the final result; that is, there is overlap in the cells which they transform by reversal.

"This method of solution overlooks the possibility that some sequence of four or more transformations, taken as a group, might have null effect. I thought about trying to solve the problem with this possibility in mind but gave it up as hopelessly expensive in view of the very low probability of such an event. I seriously doubt that there is more than one such event in the whole sequence, and am willing to settle for a suboptimal solution in that case."

The sequence SHORT contains 878 elements, and was produced by a set of six programs written in APL. It is quite possible that this list is also the solution to the problem we had in mind. In any event, Mr. Yarbrough is the richer by a Texas Instruments TI-58 calculator as the contest winner.